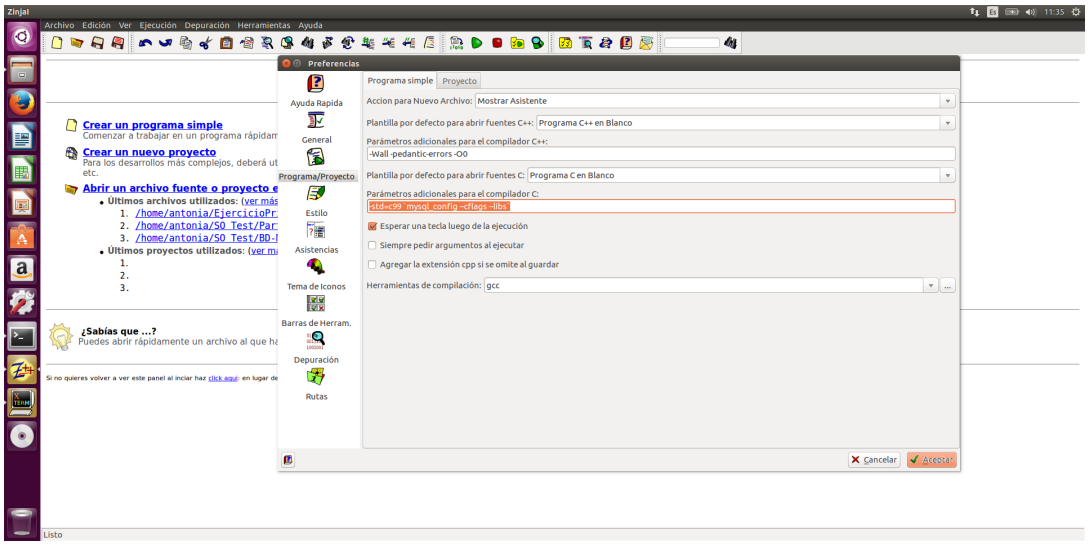
Trabajar con bases de datos MySQL

desde programas en C

Este documento muestra un ejemplo de cómo crear desde un programa en C, cómo introducir información y cómo realizar consultas SQL en una base de datos en MySQL. La base de datos será simple, con una sola entidad (tabla).

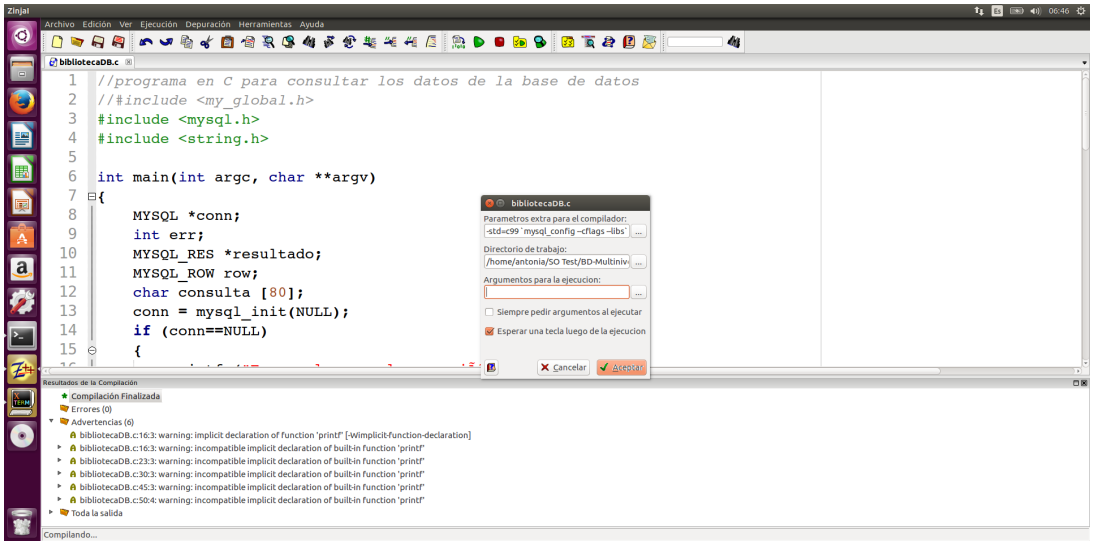
Para compilar un código en C hay que indicarle al compilador el path de los ficheros .h de MySQL y de la librería mysqlclient. En el caso de Zingal (que es el IDE – Interfaz Development Environment - que usamos) selecciona Archivo->Preferences->Program/Project y comprueba que en parámetros adiciones para el compilador tienes:

-std=c99 `mysql_config --cflags --libs`.



Podríais necesitar especificar el path y la librería anterior para un fichero concreto, por ejemplo, si el archivo de código fuente no lo has creado desde cero y lo abres con Zingal. Si es así y/o no te compila el fichero, selecciona Ejecucion -> Opciones y escribe en el recuadro de parámetros extra para el compilador de C:

-std=c99 `mysql_config --cflags --libs`.



Para compilar un programa en C que utilice las llamadas a MySQL en la máquina shiva.upc.es debe usarse el comando siguiente:

gcc -o prop prog.c `mysql_config --cflags --libs`

En las siguientes páginas web puede encontrarse información muy útil sobre el tema.

www.cyberciti.biz/tips/linux-unix-connect-mysql-c-api-program.html (conectar c y MySQL)

http://www.zetcode.com/tutorials/mysqlcapitutorial/

http://dev.mysql.com/doc/refman/5.1/en/c-api-functions.html

1. Un programa para crear una base de datos

```
//Programa en C para crear una base de datos
//Incluir esta librería para poder hacer las llamadas en shiva2.upc.es
//#include <my_global.h>
#include <mysql.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
int main(int argc, char **argv)
{
        //Conector para acceder al servidor de bases de datos
        MYSQL *conn;
        int err;
        //Creamos una conexion al servidor MYSQL
        conn = mysql_init(NULL);
        if (conn==NULL) {
                printf ("Error al crear la conexion: %u %s\n",
                                mysql_errno(conn), mysql_error(conn));
                exit (1);
        }
        //inicializar la conexion, indicando nuestras claves de acceso
        // al servidor de bases de datos (user,pass)
```

```
	conn = mysql_real_connect (conn, "localhost","root", "mysql", NULL, 0, NULL, 0);
	if (conn==NULL)
	{
		printf ("Error al inicializar la conexion: %u %s\n",
				mysql_errno(conn), mysql_error(conn));
		exit (1);
	}

	// ahora vamos a crear la base de datos, que se llamara personas

	// primero la borramos si es que ya existe (quizas porque hemos

	// hecho pruebas anteriormente

	mysql_query(conn, "drop database if exists personas;");
	err=mysql_query(conn, "create database personas;");
	if (err!=0) {
		printf ("Error al crear la base de datos %u %s\n",
				mysql_errno(conn), mysql_error(conn));
		exit (1);
	}
	//indicamos la base de datos con la que queremos trabajar

	err=mysql_query(conn, "use personas;");
	if (err!=0)
	{
		printf ("Error al crear la base de datos %u %s\n",
				mysql_errno(conn), mysql_error(conn));
		exit (1);
	}

	// creamos la tabla que define la entidad persona:

	//		un DNI (clave principal), nombre y edad

	err=mysql_query(conn,
		"CREATE TABLE personas (DNI VARCHAR(10) not null primary key, nombre
VARCHAR(25), edad int)");

	if (err!=0)
	{
		printf ("Error al definir la tabla %u %s\n",
				mysql_errno(conn), mysql_error(conn));
		exit (1);
	}
	// ahora tenemos la base de datos lista en el servidor de MySQL

	// cerrar la conexion con el servidor MYSQL

	mysql_close (conn);
	exit(0);
}
```

2. Un programa para introducir datos en la base de datos

```
// programa en C para introducir los datos en la base de datos

//Incluir esta libreria para poder hacer las llamadas en
shiva2.upc.es

//#include <my_global.h>

#include <mysql.h>

#include <string.h>
```

```
#include <stdlib.h>
#include <stdio.h>
int main(int argc, char **argv)
{
        MYSQL *conn;
        int err;
        char dni[10];
        char nombre [25];
        int edad;
        char edads [3];
        int i;
        char consulta [80];
        //Creamos una conexion al servidor MYSQL
        conn = mysql_init(NULL);
        if (conn==NULL) {
                printf ("Error al crear la conexion: %u %s\n",
                                mysql_errno(conn), mysql_error(conn));
                exit (1);
        }
        //inicializar la conexi卌n, entrando nuestras claves de
acceso y
        //el nombre de la base de datos a la que queremos
acceder
        conn = mysql_real_connect (conn, "localhost","root", "mysql", "personas",0, NULL, 0);
        if (conn==NULL) {
                printf ("Error al inicializar la conexion: %u %s\n",
                                mysql_errno(conn), mysql_error(conn));
                exit (1);
        }
        //Introduciremos en la base de datos 4 personas,
        //cuyos datos pedimos al usuario
        for (i=0; i<4; i++) {
                printf ("Escribe el DNI, nombre y edad de la persona, separados por un blanco\n");
                err = scanf ("%s %s %d", dni, nombre, &edad);
                if (err!=3) {
                        printf ("Error al introducir los datos \n");
                        exit (1);
                }
                // Ahora construimos el string con el comando SQL
```

```
                // para insertar la persona en la base. Ese string
es:
                // INSERT INTO personas VALUES ('dni',
'nombre', edad);
                strcpy (consulta, "INSERT INTO personas VALUES ('");
                //concatenamos el dni
                strcat (consulta, dni);
                strcat (consulta, "','");
                //concatenamos el nombre
                strcat (consulta, nombre);
                strcat (consulta, "',");
                //convertimos la edad en un string y lo concatenamos
                sprintf(edads, "%d", edad);
                strcat (consulta, edads);
                strcat (consulta, ");");

                printf("consulta = %s\n", consulta);
                // Ahora ya podemos realizar la insercion
                err = mysql_query(conn, consulta);
                if (err!=0) {
                        printf ("Error al introducir datos la base %u %s\n",
                                        mysql_errno(conn), mysql_error(conn));
                        exit (1);
                }
        }
        // cerrar la conexion con el servidor MYSQL
        mysql_close (conn);
        exit(0);
}
```

3. Un programa para consultar los datos de la base de datos

```
//programa en C para consultar los datos de la base de datos
//Incluir esta libreria para poder hacer las llamadas en shiva2.upc.es
//#include <my_global.h>
#include <mysql.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
int main(int argc, char **argv)
{
```

```
MYSQL *conn;
int err;
// Estructura especial para almacenar resultados de consultas
MYSQL_RES *resultado;
MYSQL_ROW row;
int edad;
char dni[10];
char consulta [80];
//Creamos una conexion al servidor MYSQL
conn = mysql_init(NULL);
if (conn==NULL) {
        printf ("Error al crear la conexi뼤n: %u %s\n",
                        mysql_errno(conn), mysql_error(conn));
        exit (1);
}
//inicializar la conexion
conn = mysql_real_connect (conn, "localhost","user", "pass", "personas",0, NULL, 0);
if (conn==NULL) {
        printf ("Error al inicializar la conexi뼤n: %u %s\n",
                        mysql_errno(conn), mysql_error(conn));
        exit (1);
}
// consulta SQL para obtener una tabla con todos los datos
// de la base de datos
err=mysql_query (conn, "SELECT * FROM personas");
if (err!=0) {
        printf ("Error al consultar datos de la base %u %s\n",
                        mysql_errno(conn), mysql_error(conn));
        exit (1);
}
//recogemos el resultado de la consulta. El resultado de la
//consulta se devuelve en una variable del tipo puntero a
//MYSQL_RES tal y como hemos declarado anteriormente.
//Se trata de una tabla virtual en memoria que es la copia
//de la tabla real en disco.
resultado = mysql_store_result (conn);
// El resultado es una estructura matricial en memoria
// en la que cada fila contiene los datos de una persona.

// Ahora obtenemos la primera fila que se almacena en una
// variable de tipo MYSQL_ROW
```

```
row = mysql_fetch_row (resultado);
// En una fila hay tantas columnas como datos tiene una
// persona. En nuestro caso hay tres columnas: dni(row[0]),
// nombre(row[1]) y edad (row[2]).
if (row == NULL)
        printf ("No se han obtenido datos en la consulta\n");
else
        while (row !=NULL) {
                // la columna 2 contiene una palabra que es la edad
                // la convertimos a entero
                edad = atoi (row[2]);
                // las columnas 0 y 1 contienen DNI y nombre
                printf ("DNI: %s, nombre: %s, edad: %d\n", row[0], row[1], edad);
                // obtenemos la siguiente fila
                row = mysql_fetch_row (resultado);
}



        // Ahora vamos a buscar el nombre de la persona cuyo DNI es uno
        // dado por el usuario
        printf ("Dame el DNI de la persona que quieres buscar\n");
        scanf ("%s", dni);
        // construimos la consulta SQL
        strcpy (consulta,"SELECT nombre FROM personas WHERE dni = '");
        strcat (consulta, dni);
        strcat (consulta,"'");
        // hacemos la consulta
        err=mysql_query (conn, consulta);
        if (err!=0) {
                printf ("Error al consultar datos de la base %u %s\n",
                                mysql_errno(conn), mysql_error(conn));
                exit (1);
        }
        //recogemos el resultado de la consulta
        resultado = mysql_store_result (conn);
        row = mysql_fetch_row (resultado);
        if (row == NULL)
                printf ("No se han obtenido datos en la consulta\n");
        else
                // El resultado debe ser una matriz con una sola fila
```

```
                    // y una columna que contiene el nombre
                    printf ("Nombre de la persona: %s\n", row[0] );
                // cerrar la conexion con el servidor MYSQL
                mysql_close (conn);
                exit(0);
}
```

4. Observación adicional

En ocasiones resulta más cómodo realizar operaciones con una base de datos directamente desde la línea de comandos (y no desde un programa en C). Por ejemplo, si queremos borrar una base de datos, podemos hacerlo ejecutando el siguiente comando:

mysql -u<nombre usuario> -p <password> DROP DATABASE <nombre de la base de datos>

De la misma manera podemos ejecutar comandos para cualquier otra consulta SQL (añadir información a la base de datos, hacer consultas, etc.).

5. Ejercicios

1. Verifica que los tres programas anteriores funcionan correctamente.
2. Escribe un programa para crear una base de datos de libros, que contenga para cada libro:
    a. El código (un número que será la clave primaria)
    b. El título (una palabra)
    c. El autor (una palabra)
    d. El precio (un número real)
3. Escribe un programa para introducir en la base de datos los datos de varios libros, que estarán almacenados en un fichero de texto (en cada línea del fichero están los datos de un libro)
4. Escribe un programa que permita realizar las siguientes consultas a la base de datos:
    - Muéstrame los datos de todos los libros
    - Muéstrame los títulos de los libros de un autor dado
    - Elimina el libro cuyo código es uno dado
    - Dame los códigos de todos los libros cuyo precio es inferior a uno dado